VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Absolvování individuální odborné praxe

Individual Professional Practise in the Company

2010 Miroslav Šustek

Souhlasím se zveřejněním této bakalářské práce dle požadavků čl. 26, odst. 9 *Studijního a zkušebního řádu pro studium v bakalářských programech VŠB-TU Ostrava*.

V Ostravě 7. května 2010 .....

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 7. května 2010 .....

Abstrakt

Tato bakalářská práce popisuje pracovní činnost vykonávanou během odborné praxe ve společnosti netdevelo s.r.o. Práci uvozuje stručná charakteristika společnosti a její historie. Hlavním cílem je pak seznámit čtenáře s druhem plněné práce. K tomu byly vybrány tři úkoly, které byly podrobně rozebrány. Přitom byl kladen důraz na objasnění postupů použitých při jejich řešení. Především uplatnění znalostí získaných při studiu bakalářského oboru Informatika a výpočetní technika.

Klíčová slova: internetový obchod, ShopSys, internetové platby, PayPal, PHP, XML dokument, CSV, datová výměna, modelování databází

Abstract

This thesis describes work which was exercised during the individual practise in the company netdevelo s.r.o. The thesis begins with characterization of the company and it's history. The main purpose is to introduce the reader with the type of performed work. Three of the assignments were chosen and described in detail. Emphasis is placed on explanation of the methods used for their solution. Especially on application of knowledge gained during the bachelor study program Computer Science and Technology.

Keywords: online shopping, ShopSys, online payments, PayPal, PHP, XML document, CSV, data exchange, database modeling

Obsah

Seznam obrázků

Seznam výpisů zdrojového kódu

1 Úvod

Během zimního a letního semestru třetího ročníku bakalářského studia jsem vykonával odbornou praxi ve společnosti netdevelo s.r.o. Tuto praxi jsem absolvoval jako alternativní možnost k vypracování klasické bakalářské práce.

Praxe ve společnosti netdevelo s.r.o. byla mou první pracovní zkušeností v oboru informatiky. Jako student bakalářského oboru *Informatika a výpočetní technika* jsem měl mnoho teoretických znalostí, které jsem mohl vlastně poprvé zúročit při práci na projektech, které neslouží pouze pro účel vlastního vzdělávání.

V kapitolách 2.1 a 2.2 je popsáno odborné zaměření firmy a také mé pracovní zařazení. Následující kapitoly 3.1, 3.2 a 3.3 obsahují popis tří úkolů, které jsem pro účely této práce zvolil ze všech zadaných úloh. U každého úkolu je uveden postup při jeho řešení. Soustředil jsem se také na teoretické znalosti, které byly potřebné pro úspěšné dokončení.

V závěrečné kapitole 4 a podkapitolách jsou uvedeny znalosti a zkušenosti, které jsem během praxe využil, a také ty, které mi scházely. Práce je zakončena souhrnem dosažených výsledků a celkovým zhodnocením praxe.

Práci jsem se snažil podat tak, aby nebylo nutné během četby studovat jiné zdroje za účelem pochopení výkladu. Výhodou pro čtenáře může být znalost programovacího jazyka PHP a databázového serveru MySQL, na kterých jsou založeny softwarové produkty společnosti netdevelo s.r.o., se kterými jsem se při praxi setkal.

2 Popis odborného zaměření firmy a pracovního zařazení

2.1 Odborné zaměření firmy

Společnost netdevelo s.r.o. byla založena v roce 2007 jako nástupnická společnost firmy ShopSys®. Zabývá se tvorbou internetových obchodů, webových portálů a prezentací, internetový marketingem a provozem serverů.

Hlavním produktem společnosti je platforma ShopSys®, která slouží jako základ pro vytváření internetových obchodů přizpůsobovaných konkrétním požadavkům klientů. Historie aplikace sahá už do roku 2003, kdy byly spuštěny první internetové obchody postavené na této platformě.

Společnost netdevelo s.r.o. je českou společností. V roce 2009 však vznikla dceřiná společnost netdevelo slovakia s.r.o., která působí na území Slovenské republiky a klade si stejné cíle na vývoj profesionálních internetových aplikací.

2.2 Pracovní zařazení studenta

Při příchodu do společnosti netdevelo s.r.o. jsem zpočátku nebyl zařazen do konkrétního oddělení. Dostával jsem pouze jednoduché úkoly, na kterých jsem postupně mohl poznat aplikaci ShopSys. Prošel jsem několika školeními a to od základního ovládání internetového obchodu z pohledu zákazníka a administrátora, přes školení vnitřní architektury systému až po školení firemní etiky, pro případ přímé komunikace s klienty společnosti.

Po necelých třech týdnech jsem byl zařazen na pozici programátora do oddělení Propojení s IS dodavatelů. Zde jsem pracoval především s částí aplikace ShopSys nazvanou *Dávkový import*. Dávkový import je podsystémem, který slouží k hromadnému vkládání a aktualizaci zboží v katalogu internetového obchodu.

Během praxe jsem měl také možnosti nahlédnout na práci, kterou vykonávali pracovníci jiných oddělení, jako například Oddělení testerů aplikací nebo Správa serverů.

3 Zadané úkoly a jejich řešení

3.1 Platební modul pro internetový platební systém PayPal

Cílem bylo nastudovat problematiku a vytvořit pro internetový obchod ShopSys platební modul, který umožní provádět platby pomocí internetového platebního systému PayPal.

3.1.1 Co je to PayPal

PayPal je jedním z tzv. e-commerce systémů, které umožňují zjednodušení internetových plateb a zvýšení jejich bezpečnosti. Uživatel platebního systému si nejprve vytvoří účet, který spojí s jednou či více kreditními nebo debetními kartami. Společnost PayPal pak při samotných platbách funguje jako prostředník, kdy odesilatel ani příjemce platby nemusí znát konkrétní bankovní údaje druhé strany.

3.1.2 Rozbor problematiky

Značnou částí řešení bylo samotné nastudování problematiky. PayPal je velice komplexní systém, a proto bylo nutné nejdříve zjistit, jak vlastně funguje, jaké poskytuje služby a jak tyto služby využít pro platby v internetovém obchodě.

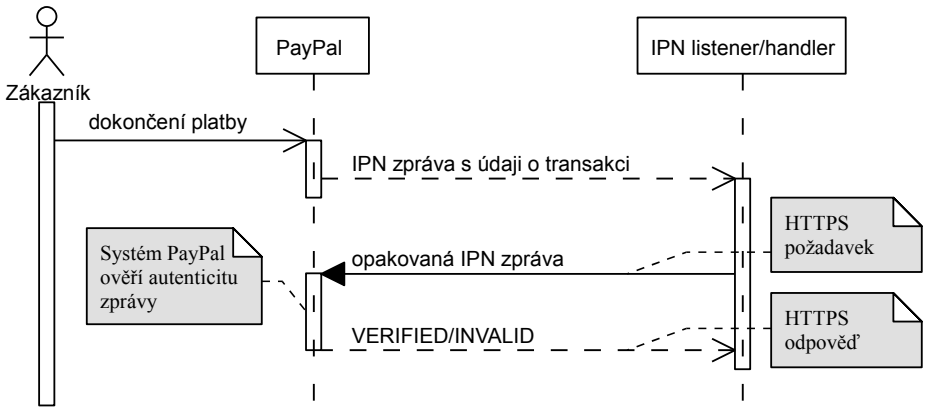
Bohužel v době, kdy jsem se touto problematikou zabýval (říjen a listopad roku 2009), ještě nebyl v provozu portál PayPal X Developer Network[4], který od počátku roku 2010 poskytuje veškeré informace potřebné k integraci služeb PayPal do informačních systémů. Čerpal jsem tedy hlavně z portálu Developer Central[1], který občas odkazoval na zastaralé či již neexistující dokumenty.

Po prostudování možností, které systém PayPal pro internetové platby nabízí, jsem dospěl k rozhodnutí, že nejvhodnější bude využít služeb Website Payments Standard a Instant Payment Notification.

3.1.2.1 Website Payments Standard Website Payments Standard je základní způsob integrace plateb pomocí speciálních tlačítek jako „Buy Now“ či „Subscribe“. Po kliknutí na takovéto tlačítko v internetovém prohlížeči je uživatel přesměrován na platební bránu systému PayPal, kde je umožněno provést platbu a to buďto po přihlášení pomocí existujícího PayPal účtu anebo přímo prostřednictvím jedné z podporovaných platebních karet bez nutnosti vlastnit PayPal účet.

Tato tlačítka mohou být tvořena HTML odkazem nebo formulářem. Součástí odkazu resp. polí formuláře jsou informace o platbě, jako je například identifikátor příjemce platby (pole business), název položky (pole item_name) či částka k úhradě (pole amount). Jako referenční seznam proměnných používaných pro Website Payments Standard jsem použil [2].

3.1.2.2 Instant Payment Notification (IPN) Jedná se o službu, která zasílá o provedených transakcích zprávy, které mohou být programově zpracovávány. Služba IPN tedy například při dokončení platby zašle na předem určenou URL adresu HTTP požadavek



Obrázek 1: Sekvenční diagram IPN komunikace

typu POST, kde jsou zaslány informace o provedené platbě. Aby byla ověřena autenticita této zprávy a potvrzeno její přijetí, příjemce zprávy (nazývaný *listener* nebo také *handler*) naváže zabezpečené spojení kanálem SSL pomocí protokolu HTTPS na adresu brány PayPal a odešle zpět přijatou zprávu. Na tento požadavek obdrží buď to odpověď VERIFIED (v případě, že je přijatá zpráva platná) nebo INVALID (v případě, že byla zpráva modifikována nebo podvržena). Tato komunikace je znázorněna sekvenčním diagramem na obrázku 1. Dokud není zpráva příjemcem potvrzena systém PayPal ji opakovaně posílá v různých časových intervalech.

3.1.3 Realizace řešení

Z Website Payments Standard jsem nevyužil žádné ze standardních tlačítek, nýbrž tzv. „Cart Upload", což je alternativa, která umožňuje na bránu PayPal odeslat libovolný obsah nákupního košíku. Dále jsem zvolil možnost vytváření souhrnné platby, kdy sice nejsou v systému PayPal zobrazeny jednotlivé položky objednávky, ale zato se omezí komplikace s různými kalkulacemi cen v systémech ShopSys a PayPal.

V takovémto případě jsou odesílány parametry cmd="_cart" a upload="1". Jako parametr item_name_1 se vyplní souhrnný název objednávky a v poli amount_1 je celková cena za objednávku. Dále je vhodné vyplnit do pole invoice jednoznačný identifikátor objednávky, aby systém PayPal mohl zamezit vícenásobné platbě za jedinou objednávku.

V aplikaci ShopSys jsem vytvořil novou PHP třídu PaymentPaypal rozšiřující třídu Payment, která je standardní součástí aplikace. Ze třídy Payment se dědí veškeré informace o objednávce (včetně položek a fakturačních údajů).

Třída PaymentPaypal pak tyto údaje připraví do formátu, který systém PayPal žádá. Bylo kupříkladu nutné přidat k základnímu číselníku států v aplikaci ShopSys kód země dle standardu ISO 3166-1 alpha-2, který je požadován pro pole country.

Pro příjem IPN zpráv jsem vytvořil novou stránku nazvanou paypal_ipn_handler, jejíž kontrolér zpracovává zprávy odesílané systémem PayPal ve formě HTTP požadavku.

Tento kontrolér ověří autenticitu zprávy a potvrdí její přijetí způsobem, který byl popsán v odstavci 3.1.2.2.

V IPN zprávách může být o transakcích přenášeno velké množství údajů (jejich kompletní popis jsem čerpal z [3]). Pro tento úkol byla však použita pouze část z nich. Podle údaje receiver_email či receiver_id se ověří, zda je uveden správný příjemce platby. Dále musí být v poli payment_status hodnota Completed, která signalizuje, že platba byla provedena a přijata. Nakonec, pokud souhlasí měna, ve které byla platba provedena (mc_currency), a částka platby (mc_gross), je v aplikaci ShopSys objednávka, identifikovaná prostřednictvím údaje invoice, označena za zaplacenou.

Pokud není přijetí IPN zprávy potvrzeno, systém PayPal tuto zprávu opakuje. Aby nedošlo k vícenásobnému zpracování jediné zprávy, jsou veškeré přijaté zprávy ukládány do tabulky vytvořené v databázi. Duplicita zpráv se pak kontroluje podle identifikátoru txn_id, který je součástí každé IPN zprávy. Pro odstranění problému se souběhem jsem, po konzultaci s kolegy s hlubšími znalostmi databázového systému MySQL, použil explicitní uzamykání této tabulky.

3.2 Převodní můstky exportu sortimentu společnosti VIVANTIS a.s.

Úkolem bylo vytvoření tzv. *můstků*, které budou převádět exportní XML soubory sortimentu společnosti VIVANTIS a.s. do formátu, který přijímá Dávkový import aplikace ShopSys – tedy formátu CSV.

3.2.1 Teoretický základ

Internetový obchod ShopSys nabízí možnost hromadného importu a aktualizace katalogu produktů. K tomuto slouží součást nazvaná *Dávkový import*.

Dávkový import přijímá jako vstup soubor ve formátu CSV, kde každý jednotlivý řádek reprezentuje informace o jednom produktu, jako například název produktu, cenu, katalogové číslo, název výrobce apod. Stručný popis samotného formátu CSV následuje v odstavci 3.2.1.1.

Mnoho dodavatelů však poskytuje svým partnerům katalog sortimentu ve formátu XML. Nastává zde tedy potřeba převést tato XML data do formátu CSV, aby je bylo možné do internetového obchodu naimportovat. Přesně tuto funkci zastává můstek.

3.2.1.1 Souborový formát CSV Formát CSV (celým názvem *Comma-Separated Values*) je textový formát sloužící k ukládání tabulkových dat. Pro tento formát prozatím neexistuje žádný obecný standard. Pro internetovou komunikaci byl organizací *IETF* vytvořen informativní dokument RFC4180 [5]. Formát CSV používaný v aplikaci ShopSys se až na některé výjimky drží formátu popsaného v tomto dokumentu.

Každý záznam (řádek) tabulky je v souboru uložen jako jeden řádek[1]. Sloupce tabulky jsou pak odděleny znakem *středník* (;). Pokud se má v textu buňky vyskytnout tento

[1] Mimo uvedené výjimky, kdy může jeden záznam obsadit více řádků souboru.

znak oddělovače nebo je text víceřádkový, musí být uzavřen do *dvojitých uvozovek* (”). Je-li znak dvojitá uvozovka obsažen v textu buňky, musí být tento znak zdvojen (tedy dvě dvojité uvozovky). Jako znaková sada se pro soubor používá kódová stránka CP1250.

Ve výpisu 1 je ukázka souboru CSV, který ukládá data shodná s těmi, která jsou vyobrazená v tabulce 1. Pro srovnání uvádím ve výpisu 2 příklad, jak by mohla vypadat reprezentace těchto dat pomocí dokumentu jazyka XML.

```
Katalogové číslo;Název produktu;Cena;Popis
KM-002-58;”Šroub M2; 5x8”;3,00;Jednořádkový popis
FDD35;”3,5””disketa”;9,90;”Víceřádkový↩
popis produktu”
```

Výpis 1: Ukázka souboru ve formátu CSV používaném v aplikaci ShopSys

Katalogové číslo	Název produktu	Cena	Popis
KM-002-58	Šroub M2; 5x8	3,00	Jednořádkový popis
FDD35	3,5”disketa	9,90	Víceřádkový↩ popis produktu

Tabulka 1: Referenční data

3.2.2 Možnosti zpracování XML dokumentů v jazyce PHP

Jazyk PHP poskytuje několik způsobů, jak načítat a analyzovat XML dokumenty. Jednou z možností je využít rozšíření XML Parser. Toto rozšíření poskytuje podporu pro zpracování XML dat založené na událostech (pro stejný přístup se také používá označení *SAX – Simple API for XML*). Jsou definovány tři typy událostí:

- přečtena počáteční značka (například <POLOZKA>),
- přečtena koncová značka (například </POLOZKA>),
- přečtena vnitřní data elementu (například FDD35).

Pro každou z těchto událostí se zaregistruje jedna uživatelská funkce, která událost zpracovává. Dokument je při zpracování čten sekvenčně a při zmíněných událostech je vždy zavolána patřičná funkce.

Tento přístup byl používán v původních převodních můstcích určených pro starou datovou výměnu pomocí XML. Ve chvíli, kdy je krom pouhého přenosu hodnot z XML do CSV potřeba v můstku provádět i nějaké operace nad daty, popřípadě je stromová struktura komplikovanější, stává se tento způsob zpracování poněkud nepohodlný. Programátor se kupříkladu musí sám starat o uchovávání informace v jakém kontextu (resp. v jaké části XML stromu) se v dokumentu právě nachází a funkce pro zpracování událostí se ve spoustě případů stanou jedním velkým a nepřehledným blokem *switch-case*.

Mnohem komfortnější způsob práce s XML dokumenty v jazyce PHP (od verze 5) nabízí rozšíření SimpleXML a jeho třída SimpleXMLElement. Na rozdíl od XMLParser toto

rozšíření umožňuje pracovat s XML jako se stromem, konkrétně stromem objektů typu SimpleXMLElement. Navíc je podporována určitá podmnožina dotazovacího jazyka *XPath*.

Cílem SimpleXML je zajistit jednoduchý přístup k XML. Objekty SimpleXMLElement se řídí následujícími čtyřmi pravidly:

1. Prostřednictvím vlastností lze přistupovat k iterátorům elementů XML.
2. Prostřednictvím číselných indexů lze přistupovat k elementům XML.
3. Prostřednictvím nečíselných indexů lze přistupovat k atributům elementů.
4. Textová data se získají přetypováním na datový typ string.

Jednoduchost použití se pokusím prezentovat ve výpisu 3. Rozšíření SimpleXML je zde použito pro načtení souboru *katalog.xml*, který obsahuje XML dokument z výpisu 2. Pomocí *XPath* dotazu jsou vybrány všechny položky, které mají cenu vyšší něž 5. Poté jsou výsledné elementy POLOZKA procházeny a z každého je vypsán obsah vnořeného elementu NAZEV.

```
<?xml version="1.0" encoding="utf-8"?>
<KATALOG>
   <POLOZKA>
      <KOD>KM-002-58</KOD>
      <NAZEV>Šroub M2; 5x8</NAZEV>
      <CENA>3.00</CENA>
      <POPIS>Jednořádkový popis</POPIS>
   </POLOZKA>
   <POLOZKA>
      <KOD>FDD35</KOD>
      <NAZEV>3,5" disketa</NAZEV>
      <CENA>9.90</CENA>
      <POPIS>Víceřádkový↩
popis produktu</POPIS>
   </POLOZKA>
</KATALOG>
```

Výpis 2: Ukázka souboru ve formátu XML

```
<?php
   // načtení XML dokumentu ze souboru
   $xml = new SimpleXMLElement("katalog.xml", 0, true);

   // získání pole objektů SimpleXMLElement reprezentující
   // elementy POLOZKA s cenou vyšší než 5
   $polozky = $xml->xpath("/KATALOG/POLOZKA[CENA>5]");

   // výpis názvů vybraných položek
   foreach ($polozky as $polozka) {
      echo (string)$polozka->NAZEV . "\n";
   }
?>
```

Výpis 3: Práce s XML dokumentem pomocí PHP rozšíření SimpleXML

Oproti zpracování pomocí XML Reader je velkou výhodou, že jsou informace o jednotlivých položkách dostupné najednou, a nikoliv postupně, když se na ně narazí při sekvenčním čtení.

Objektový přístup má však také své nevýhody. Hlavním neduhem jsou vyšší nároky na paměťový prostor. V krajním případě musí být celý dokument uložen v paměti jako struktura objektů a požadavky na paměť mohou dosáhnout i několikanásobku velikosti zpracovávaného dokumentu.

3.2.3 Realizace řešení

Společnost VIVANTIS a.s. měla, s přechodem na nový formát datové výměny, za cíl vytvořit univerzální strukturu pro všechny poskytované XML exporty. Téměř každý sortiment (*Hodinky.cz*, *Krása.cz*, *Parfémy.cz*, *Prozdraví.cz* a *Šperky.cz*) však obsahoval nějaké své specifikum, a proto jsem raději opět vytvořil pro každý z exportů zvláštní převodní můstek. Úmyslem bylo vytvořit přehlednější kód, který bude snadněji přizpůsobitelný konkrétním požadavkům klientů.

Pro zpracování XML dokumentů jsem použil rozšíření SimpleXML, které bylo blíže popsáno v kapitole 3.2.2. Pokusně jsem ověřil, že paměťové nároky při zpracování jsou přijatelné. Můstky jsou navíc Dávkovým importem spouštěny pouze několikrát za den, většinou v nočních hodinách, kdy probíhá aktualizace katalogu.

Pro zápis dat ve formátu CSV jsem použil vestavěnou PHP funkci fputcsv. Této funkci se jako parametry předají prostředek, do kterého má být zapisováno, a pole řetězců, které reprezentuje buňky jednoho řádku tabulky. Je možné také zadat oddělovač, který má být v CSV použit, a znak ohraničující víceřádkové buňky.

Převodní můstky jsou vytvořeny jako samostatné PHP skripty. Při spuštění tohoto skriptu prostřednictvím webového serveru dojde nejprve ke stažení zdrojového XML souboru, poté se vygeneruje výstupní CSV soubor, který se jednak zapíše na disk (pro možnost následného přezkoumání), a také je výstup poslán jako tělo odpovědi na HTTP požadavek.

Na URL adresu můstku je možné přímo nastavit tzv. *periodickou dávku*. Dávkový import pak tento vstup zpracovává automaticky pravidelně v určených časových intervalech. V případě chyby při práci můstku, například pokud se nepodařilo stáhnout zdrojový soubor nebo tento XML dokument neprošel kontrolou validity, je v hlavičce odpovědi navrácen chybový stav 503 – Service unavailable a periodická dávka není provedena.

3.3 Import parametrů produktů od více dodavatelů

Cílem bylo vytvořit řešení, které umožní do jediného internetového obchodu importovat od více dodavatelů technické parametry k produktům a to tak, aby bylo umožněno jednotné vyhledávání dle parametrů.

3.3.1 Úvod do problematiky

Technické parametry produktů mají v internetovém obchodě jednak informativní charakter, kdy se zobrazují na stránce s detailem produktu, ale také slouží pro vyhledávání v katalogu.

Dávkový import, který je určen pro hromadné vkládání a aktualizaci sortimentu, disponuje možností importovat k produktům také parametry. Je zde ovšem jisté omezení v podobě nutnosti vytvořit tzv. *šablonu parametrů*, která definuje seznam parametrů, které se budou k produktům importovat. Například pro televizory by takováto šablona mohla obsahovat parametry: *výška*, *šířka*, *uhlopříčka obrazovky* a *příkon*. Šablona může být pouze jedna pro jeden importovaný soubor.

Pro případy, kdy je importován široký sortiment, by seznam parametrů v šabloně musel zahrnovat všechny parametry od každého druhu sortimentu. V případě, kdy není výčet parametrů v importovaném souboru pevně stanoven, se potom Dávkový import pro přenos parametrů použít nedá. K tomuto účelu jsou v aplikaci ShopSys vytvářeny speciální služby, které slouží výhradně pro import parametrů produktů z konkrétního zdroje (většinou přímo od dodavatele sortimentu).

Problém nastal, když došlo k importu parametrů pro stejný sortiment od různých dodavatelů. Docházelo k tomu, že dodavatelé používali pro významově stejné parametry různé názvy a lišily se i hodnoty těchto parametrů. A tak, když opět použiji příklad s televizory, byly při vyhledávání nabízeny parametry *úhlopříčka obrazovky* a *úhlopříčka*. Hodnoty pak byly například v jednom případě pouze číselné a ve druhém případě doplněny o zkratku „cm“.

3.3.2 Postup řešení

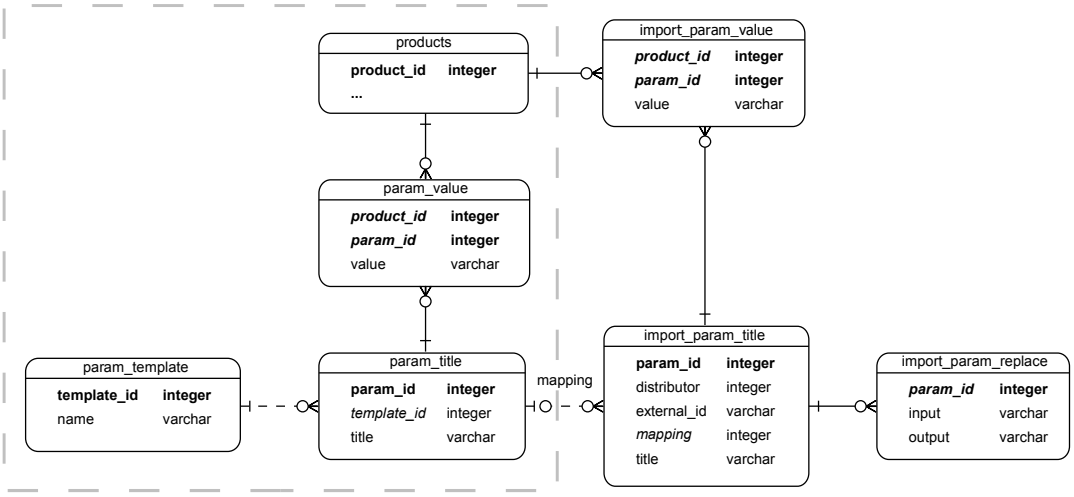
3.3.2.1 Analýza stávajícího systému Nejprve jsem se musel zorientovat v tom, jak se s parametry produktů pracuje ve stávajícím systému.

- Parametry jsou seskupovány do šablon parametrů.
- Produkty mají k parametrům přiřazeny hodnoty.
- Seznam parametrů nabízených při vyhledávání je dán šablonou.

Část původní databázové struktury, která se týká technických parametrů produktů, je zobrazena v šedě orámované části E-R diagramu na obrázku 2.

3.3.2.2 Návrh úprav Importované parametry a jejich hodnoty není možné ukládat do existujících tabulek právě kvůli výskytu významově stejných parametrů s různými názvy. Pro importované parametry a hodnoty byly vytvořeny tabulky import_param_title a import_param_value.

Abych zachoval stávající přístup parametrického vyhledávání dle šablony parametrů, byla má myšlenka taková, že parametrům v šablonách se přiřadí importované parametry.



Obrázek 2: E-R diagram databázové struktury týkající se parametrů

Za parametrem v šabloně se pak budou skrývat hodnoty od všech přiřazených importovaných parametrů. K přiřazení importovaných parametrů k parametrům v existující šabloně slouží vazba mapping.

Pro sjednocení hodnot parametrů jsem zvolil řešení, kdy se pro každý importovaný parametr vytvoří množina nahrazení hodnot. Znamená to, že definuji pro vstupní hodnotu parametru jinou hodnotu, kterou bude ta původní nahrazena. Proto byla vytvořena tabulka import_param_replace.

Zbývalo vyřešit otázku, jakou hodnotu preferovat, pokud bude pro jeden parametr existovat jak hodnota přidaná ručně, tak importovaná hodnota. Ze strany zadavatele padlo rozhodnutí, že takovém případě bude upřednostňována ručně vložená hodnota.

Výslednou strukturu databáze vyobrazuje kompletní E-R diagram na obrázku 2.

3.3.2.3 Úpravy aplikační logiky Na straně služeb pro aktualizaci parametrů byly nutné pouze minimální úpravy. Parametry a hodnoty se vkládají do jiných tabulek (s prefixem "import_"). Dále musí být vyplněn identifikátor dodavatele (distributor) a volitelně může být použit atribut external_id . Jelikož většina zdrojových souborů poskytuje vlastní identifikátor parametru, může být uložen právě do tohoto atributu, aby mohl být název parametru později aktualizován (při přejmenování parametru ve zdrojovém souboru tak nedojde ke smazání původního a vytvoření nového parametru).

Dále bylo potřeba upravit SQL dotaz, který provádí samotné filtrování. Struktura tohoto dotazu je znázorněna ve výpisu 4. Pokud je vyhledáváno podle více parametrů najednou, bude ve výsledném SQL dotazu několik konstrukcí „WHERE P.product_id IN“.

Jako nejjednodušší cestu jsem viděl vytvoření pohledu, který by bral v úvahu nastavené sjednocení parametrů a hodnot a zároveň by skloubil dohromady importované i ručně vložené parametry. Vytvořil jsem tedy pohled nazvaný param_value_view, jehož

definice je ve výpisu 5. V SQL dotazu pro parametrické vyhledávání pak stačilo nahradit tabulku param_value tímto pohledem.

```
SELECT ... FROM products P
WHERE P.product_id IN (
    SELECT product_id FROM param_value
    WHERE param_id = ? AND value = ?
)
```

Výpis 4: Struktura SQL dotazu pro filtrování dle parametrů

Toto řešení by však samo o sobě nebylo příliš efektivní. Nad pohledem, který provádí operaci UNION, totiž nelze vyhledávat pomocí indexu, tak jako v původní tabulce. Je veliká škoda, že databázový systém MySQL nenabízí možnost vytvořit *indexovaný pohled* popřípadě *materializovaný pohled*, které jsou podporovány produkty Microsoft SQL Server respektive Oracle Database.

```
CREATE param_value_view AS
SELECT product_id, param_id, value FROM param_value
UNION
SELECT IPV.product_id, PT.param_id, IFNULL(IPR.output, IPV.value)
FROM import_param_value IPV
JOIN import_param_title IPT ON IPV.param_id = IPT.param_id
JOIN param_title PT ON IPT.mapping = PT.param_id
LEFT JOIN import_param_replace IPR ON IPT.param_id = IPR.param_id
    AND IPV.value = IPR.input
WHERE NOT EXISTS (
    SELECT 1 FROM param_value PV
    WHERE PV.product_id = IPV.product_id AND PV.param_id = PT.param_id
)
```

Výpis 5: SQL skript definující pohled pro importované parametry

Alespoň částečně se dá tato funkcionalita nahradit pomocí klasické tabulky. Tato tabulka se pak musí pomocí dotazu, který by byl použit pro pohled, přegenerovat při každé změně ve zdrojových tabulkách. Nad klasickou tabulkou pak samozřejmě index vytvořit lze. Tento postup byl nakonec použit i ve výsledném řešení. Za cenu zvýšené režie při importování parametrů a při změnách nastavení přiřazení a nahrazování byla získána efektivita dotazování srovnatelná s původním řešením.

Další úpravy bylo potřeba provést v administračním rozhraní internetového obchodu. Návrh a implementaci těchto úprav, na základě vytvořeného modelu databázové struktury a specifikace požadavků, již provedli pracovníci Oddělení implementace.

4 Závěr

4.1 Znalosti a dovednosti uplatněné v průběhu praxe

Velkým přínosem byl předmět *Úvod do softwarového inženýrství*, který docela podstatně změnil můj pohled na vývoj software. Během absolvování praxe jsem se utvrdil v tom, že podstatnou části vývoje softwarových produktů jsou analýza, návrh a modelování. Pokud jsou tyto kroky na začátku provedeny kvalitně, samotná implementace řešení se stává znatelně jednodušší a přímočařejší. Problémům, které se nevyřeší už při návrhu, je tak jako tak nutné čelit, ovšem v pozdějších fázích vývoje to stojí podstatně více úsilí a času.

Nepostradatelné pro mě byly znalosti týkající se databází a databázových systémů. Kurzy *Teorie zpracování dat* a *Databázové a informační systémy* mi pomohly zorientovat se v databázové problematice a ukázaly mi metody pro návrh a modelování databází.

Ocenil jsem také znalosti z předmětu *Počítačové sítě*, především ty z okruhu internetových protokolů TCP/IP a HTTP.

V kurzu *Java technologie* jsem získal základní přehled o jazyce XML a metodách zpracování XML dokumentů.

4.2 Znalosti či dovednosti scházející v průběhu praxe

Vývoj informačních systémů nebýval zrovna středem mých zájmů, proto jsem měl pouze několik málo teoretických znalostí. Navíc kurz *Tvorba informačních systémů* byl zařazen až v letním semestru posledního ročníku.

S jazykem PHP jsem měl také spíše jen letmé zkušenosti. Jeho syntaxe a požívané konstrukce jsou však podobné programovacím jazykům, které jsem znal (C, C#, Java), a tak jsem se s ním naučil pracovat poměrně rychle.

4.3 Výsledky dosažené v průběhu odborné praxe a celkové zhodnocení

Platební modul pro systém PayPal byl již nasazen do ostrého provozu na dvou internetových obchodech.

Všichni klienti, kteří používali datovou výměnu se společností VIVANTIS a.s., museli postupně přejít na nové převodní můstky, neboť starou strukturu exportních souborů již společnost VIVANTIS a.s. přestala používat.

Praktické zkušenosti nabyté během této praxe jsou pro mne velice hodnotné. Zjistil jsem, jak probíhá vývoj aplikací ve větším kolektivu. Poznal cestu, kterou plynou požadavky od zadavatele přes projektového konzultanta a analytika až do konečné specifikace pro programátora.

Také jsem poznal, že v praxi dokonalé splnění specifikace ještě nemusí znamenat úplnou spokojenost na straně zadavatele, který mnohdy získá přesnou představu o svých požadavcích až když vidí nějaký výsledek.

Řekl bych, že je dobré si již při studiu prakticky vyzkoušet některou z možných budoucích profesí a získat tak cenné zkušenosti, které se na školách nelze naučit.

5 Reference

[1] *Developer Central - PayPal* [online]. [cit. listopad 2009].
<https://cms.paypal.com/us/cgi-bin/?cmd=_render-content&content_ID=developer/home>

[2] *HTML Variables for Website Payments Standard* [online]. [cit. 2010-04-24]
<https://cms.paypal.com/us/cgi-bin/?&cmd=_render-content&content_ID=developer/e_howto_html_Appx_websitestandard_htmlvariables>

[3] *IPN and PDT Variables* [online]. [cit. 2010-04-24]
<https://cms.paypal.com/us/cgi-bin/?cmd=_render-content&content_ID=developer/e_howto_html_IPNandPDTVariables>

[4] *PayPal X Developer Network* [online]. [cit. 2010-04-24].
<https://www.x.com/>

[5] SHAFRANOVICH, Y., *Common Format and MIME Type for Comma-Separated Values (CSV) Files* [online]. Říjen 2005 [cit. 2010-04-26]. <http://tools.ietf.org/rfc/rfc4180.txt>